Applications of the Conti-Traverso Algorithm for Traveling Salesman Problems (Mathematical Optimization Theory and its Algorithm)

AUTHOR(S):

Ito, Masashi; Hirabayashi, Ryuichi

数理解析研究所講究録 1241 巻 2001 年 75-83

Applications of the Conti-Traverso Algorithm for Traveling Salesman Problems

伊藤 雅史 (M. Ito)[1], 平林 隆一 (R. Hirabayashi)[2]
東京理科大学工学部 (Science University of Tokyo)

1 Backgrounds and objectives

Groebner bases for ideals in polynomial rings were introduced in 1965 by B. Buchberger. He also developed the fundamental algorithms which was called "*the Buchberger algorithm*" to compute a Groebner bases from a generating set of ideals [6]. The theory of Groebner bases and this algorithm provide the foundation for many algorithms in both algebraic geometry and commutative algebra, and they are applied as a method of solving systems of polynomial equations, and so on. Groebner bases are very powerfull tool, but it is known that to compute them is extremely time-consuming.

In 1991, Conti and Traverso proposed an algorithm for integer programming problems (IP) using the Groebner base [8]. This algorithm makes good use of toric ideals, a special class of polynomial ideals. Thanks to their algorithm, the theory of the Groebner basis has come to be one of the interplay of computational commutative algebra and optimizations [18][20][21][23][22]. Our objectives are;

(i) to measure the efficiency of Groebner bases for combinatorial optimization problems, and

(ii) to analyze the structure of combinatorial optimization problems from the viwe of toric ideals and Groebner bases.

In this paper, we developed an algorithm for traveling salesman problem (TSP), in which we combined the Conti-Traverso algorithm and branch and bound method.

2 Preliminaries and Previous Works

2.1 The Traveling Salesman Problem(TSP)

The TSP is one of the most difficult combinatorial problems. It is already known that the problem belongs to the class NP-complete[17]. Basic references are [3] [5], and [15].

We consider the TSP on complete undirected graphs with d nodes, denoted by K_d. Let $N = \{1, \ldots, n\}$ be the node set, $E = \{\{i,j\} : 1 \leq i < j \leq n\}$ be the edge set, and $c_{ij} \in \mathbb{R}^n_{\geq 0}$ be the cost of edge $\{i,j\}$. For each edge $\{i,j\} \in E$, we introduce a variable $x_{ij} \in \{0,1\}$. In our formulation, there are 3 type of constraints;

(i) Degree Constraints: For each vertex $i \in N$, just two edges incident with i must be chosen,

(ii) Subtour Elimination: for each subtour $C \subset E$, the cardinality of the set $\{\{i,j\} \in E : x_{ij} = 1\}$ is less than $|C|$, where $|\cdot|$ means the cardinality, and

(iii) 0-1 Constraints: for each edge $\{i,j\} \in E$, $x_{ij} \in \{0,1\}$.

Let $\mathcal{C} \subset 2^E$ be the set of all subtours. Then TSP is represented as follows.

$$
\begin{array}{lll}
\text{minimize} & \sum_{\{i,j\} \in E} c_{ij} x_{ij}, & \\
\text{subject to} & \sum_{j\,:\,\{i,j\} \in E} x_{ij} = 2, & \forall i \in N, \\
& \sum_{\{i,j\} \in C} x_{ij} \leq |C| - 1, & \forall C \in \mathcal{C}, \\
& x_{ij} \in \{0,1\}, & \forall \{i,j\} \in E.
\end{array}
$$

[1] masafumi@ms.kagu.sut.ac.jp
[2] hira@ms.kagu.sut.ac.jp

2.2 Groebner Bases

In this subsection, we summarize some basic definitions of the theory of Groebner basis for ideals. Basic references are [1], [2], and [9].

Let K be an arbitrary field. Denote by K[y] the polynomial ring with y where y consists of n variables $y_1, \ldots, y_n$, *i.e.*, $\mathbf{y} = (y_1, \ldots, y_n)$.

A subset $I \subset \mathrm{K}[\mathbf{y}]$ is an ideal on K[y] if it satisfies: (i) $0 \in I$, (ii) if $f, g \in I$, then $f + g \in I$, and (iii) if $f \in I$ and $h \in \mathrm{K}[\mathbf{y}]$, then the product $hf \in I$. Let $f_1, \ldots, f_s$ be polynomials in K[y] and denote

$$\langle f_1, \ldots, f_s \rangle \stackrel{\text{def}}{=} \{ \sum_{i=1}^{s} h_i f_i \ : \ h_1, \ldots, h_s \in \mathrm{K}[\mathbf{y}] \}.$$

It is easy to show that $\langle f_1, \ldots, f_s \rangle$ is an ideal. We call such set $\{f_1, \ldots, f_s\}$ a *generating set* or a *basis*. Furthermore, $\langle f_1, \ldots, f_s \rangle$ called the ideal generated by $f_1, \ldots, f_s$. It is known as the Hilbert basis theorem that each polynomial ideal are finitely generated.

A term order on K[y] is any relation $\succ$ on $\mathbf{N}^n$, satisfying: (i) $\succ$ is a total ordering on $\mathbf{N}^n$, (ii) if $\alpha, \beta, \gamma \in \mathbf{N}^n$ and $\alpha \succ \beta$, then $\alpha + \gamma \succ \beta + \gamma$, and (iii) $\mathbf{0}$ is the minimal element w.r.t. $\succ$. As the examples of the term order, the lexicographic order (lex), graded lex order, and graded reverse lex order are well known.

Now we fix a term order $\succ$ on $\mathbf{N}^n$. Let $I \subset \mathrm{K}[\mathbf{y}]$ be an ideal other than $\{0\}$. Then for all polynomial f in K[y], the greatest term with respect to $\succ$ is uniquely defined. Call this term the *initial term* of f *w.r.t.* $\succ$, and denote by $\mathrm{in}_{\succ}(f)$. Furthermore, we define the initial ideal $\mathrm{in}_{\succ}(I)$ of I *w.r.t.* $\succ$ as follow:

$$\mathrm{in}_{\succ}(I) \stackrel{\text{def}}{=} \langle \mathrm{in}_{\succ}(f) \ : \ f \in I \rangle.$$

The Groebner basis is defined as follows.

Definition 2.1. (Groebner basis). *A finite subset* $\{g_1, \ldots, g_t\}$ *of* I *is a Groebner basis of* I *w.r.t* $\succ$, *if it satisfies,* $\langle \mathrm{in}_{\succ}(g_1), \ldots, \mathrm{in}_{\succ}(g_t) \rangle = \mathrm{in}_{\succ}(I)$.

For a pair of monomials $\mathbf{y}^{\alpha}$ and $\mathbf{y}^{\beta}$, we say $\mathbf{y}^{\alpha}$ is *divisible* by $\mathbf{y}^{\beta}$ if, in the vector difference $\alpha - \beta$, all entries are nonnegative. It is known that for all ideals except $\{0\}$, there exists a Groebner basis.

Let f be an arbitrary polynomial in K[y] and I be an ideal. Then the remainder on division of f by a Groebner basis of I is unique. The remainder called the normal form of f *w.r.t.* $\succ$, and denoted by $\mathrm{nf}_{\succ}(f)$.

A Groebner basis $G_{\succ}(I)$ is reduced, if satisfying: (i) For all $g \in G_{\succ}(I)$, the coefficient of $\mathrm{in}_{\succ}(g)$ is 1, and (ii)] for all $g, h \in G_{\succ}(I)$, any term of h is not divisible by $\mathrm{in}_{\succ}(g)$. The reduced Groebner basis uniquely exists for all ideal $I \neq \{0\}$ and any term order $\succ$.

An Groebner basis can be computed by the *Buchberger algorithm*. It is known that Buchberger algorithm is extremely time-consuming. There are several studies about speedup of this algorithm [6], [12], [24], but even with the best currently known versions of the algorithm, computation of a Groebner basis often takes a tremendously long time and storage space.

2.3 Toric Ideals

We shall explain the toric ideal, a special class of ideals in K[y]. Basic references are [1], [11] and [19] Let $\mathbf{A}$ be a $d \times n$ matrix of $\mathbf{Z}^n$, where $\mathbf{Z}$ is set of all integers. Denote the ith column vector of $\mathbf{A}$ by $\mathbf{a}_i$. Each vector $\mathbf{a}_i$ is identified with a monomial $\mathbf{t}^{\mathbf{a}_i}$ in the *Laurent polynomial ring*:

$$\mathrm{K}[\mathbf{t}^{\pm 1}] \stackrel{\text{def}}{=} \mathrm{K}[t_1, \ldots, t_d, t_1^{-1}, \ldots, t_d^{-1}].$$

Consider the semigroup homomorphism π and the ring homomorphism ϕ;

$$\pi : \mathbb{N}^n \to \mathbb{Z}^d, \mathbf{u} = (u_1, \ldots, u_n) \mapsto u_1\mathbf{a}_1 + \cdots + u_n\mathbf{a}_n,$$
$$\phi : \mathrm{K}[\mathbf{y}] \to \mathrm{K}[\mathbf{t}^{\pm 1}], y_i \mapsto \mathbf{t}^{\mathbf{a}_i}.$$

The toric ideal of $\mathbf{A}$ is defined as the kernel of ϕ. For all vector $\mathbf{u} \in \mathbb{Z}^n$, we represent $\mathbf{u} = \mathbf{u}^+ - \mathbf{u}^-$ with nonnegative vectors $\mathbf{u}^+, \mathbf{u}^-$. We denote the integral kernel of π by $\mathrm{Ker}_{\mathbf{Z}}(\pi)$. Then, the toric ideal $I_{\mathbf{A}}$ can be written as following:

$$I_{\mathbf{A}} = \langle \mathbf{y}^{\mathbf{u}^+} - \mathbf{y}^{\mathbf{u}^-} \ : \ \mathbf{u} \in \mathrm{Ker}_{\mathbf{Z}}(\pi) \rangle$$

Now we talk about computation of toric ideal. The method we explain is based on the Implicitization Theorem and the Elimination Theorem, detail is in Chapter 3 of [9].

Algorithm 2.2. (Compute the reduced Groebner basis of toric ideals [19]).

STEP 1: *Introduce* $n + d + 1$ *variables* $t_0, t_1, \ldots, t_d, y_1, \ldots, y_n$.
Let $\succ$ *be any elimination term order*
such that $\{\mathbf{t}\} \succ \{\mathbf{y}\}$.

STEP 2: *Compute the reduced Groebner basis* G *of the ideal*

$$\langle t_0 t_1 \cdots t_d - 1, y_1 \cdot \mathbf{t}^{\mathbf{a}_i^-} - \mathbf{t}^{\mathbf{a}_i^+}, \ i = 1, \ldots, n \rangle.$$

STEP3 : *The set* $G \cap \mathrm{K}[\mathbf{y}]$ *is the reduced Groebner basis of* $I_{\mathbf{A}}$
w.r.t. restricted order of $\succ$ *on* $\mathrm{K}[\mathbf{y}]$.

There are several methods of the Groebner basis computation for toric ideals [4], [16], [18]. To compute a Groebner basis $\mathrm{G}_{\succ_{\mathbf{ty}}}(J)$, remark the following fact.

Remark 2.3. *Denote* $F = \{x_i \mathbf{t}^{\mathbf{a}_i^-} - \mathbf{t}_i^+ \ : \ i = 1, \ldots, n\}$. *Then* F *is already the reduced Groebner basis of* J *w.r.t. an elimination term order* $\succ_{\mathbf{yt}}$ *such that* $\{\mathbf{y}\} \succ_{\mathbf{yt}} \{\mathbf{t}\}$.

Thanks to this fact, $\mathrm{G}_{\succ_{\mathbf{ty}}}(J)$ can be obtained by applying a kind of basis conversion algorithm, that converts a Groebner basis *w.r.t.* $\succ_{\mathbf{yt}}$ into a Groebner basis *w.r.t.* $\succ_{\mathbf{ty}}$. Several algorithm are already known [7], [13], [14], [7], but in this case, the Groebner Walk algorithm is suitable because of its geometric features. On the polynomial ring $\mathrm{K}[\mathbf{t}, \mathbf{y}]$, set the variable ordering to $t_1 > \cdots > t_d > y_1 > \cdots > y_n$. Then the Groebner Walk algorithm will be done in the vector space $\mathbb{R}^{d+n}$. The departure $\boldsymbol{\sigma}$ and the destination $\boldsymbol{\tau}$ are, for example,

$$\boldsymbol{\sigma} = \underbrace{(0, \ldots, 0}_{d}, \underbrace{1, \ldots, 1)}_{n}, \text{and}$$
$$\boldsymbol{\tau} = \underbrace{(1, \ldots, 1}_{d}, \underbrace{0, \ldots, 0)}_{n}.$$

With respect to the cost vector $\boldsymbol{\sigma}$ (*resp.* $\boldsymbol{\tau}$), a term which contains at least one of y_i (*resp.* t_j) is always larger than a term which does not contain any y_i (*resp.* t_j). Furthermore, fix the elimination term order $\succ_{\mathbf{ty}}$ as the tie-breaker. Then the desired Groebner basis can be obtained by the Groebner Walk algorithm.

2.4 The Conti-Traverso Algorithm

In this subsection, we talk about the Conti-Traverso algorithm for an integer programming [8]. There are many method for solving integer programming problems[17]. But the Conti-Traverso algorithm is a purely algebraic, and its basic idea and the implementation are simple.

Let $\mathbf{A}$ be a $d \times n$ integral matrix with full row rank. Consider following *integer programming problem* (IPP)

$$\mathrm{IP}_{\mathbf{A},\mathbf{c}}(\mathbf{b}) \stackrel{\text{def}}{=} \text{minimize}\{\mathbf{c} \cdot \mathbf{x} \ : \ \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}.$$

The *coefficient matrix* $\mathbf{A}$ is as above, the *right hand side vector* $\mathbf{b} \in \mathbb{Z}^d$, and the *cost vector* $\mathbf{c} \in \mathbb{R}^n$. Assume that, (i) $\mathbf{b} \in \{\mathbf{Au} \ : \ \mathbf{u} \in \mathbb{N}^n\}$, and (ii) $\{\mathbf{x} \geq \mathbf{0} \ : \ \mathbf{Ax} = \mathbf{0}\} = \{\mathbf{0}\}$. The assumption (i) guarantees that $\mathrm{IP}_{\mathbf{A},\mathbf{c}}(\mathbf{b})$ is feasible. From the theory of convex polytopes, the assumption (ii) implies that $\mathrm{P}_{\mathbf{b}}$ is a polytope(bounded polyhedron), hence $\mathrm{P}_{\mathbf{b}}^I$ is again a polytope. Let $\succ$ be a term order, and $\mathbf{c} \in \mathbb{R}^n$. An order $(\mathbf{c}|\succ)$ on $\mathbb{N}^n$ is defined as follow; comparing by the inner product with $\mathbf{c}$ and use $\succ$ as the tie-breaker. If $\mathbf{c}$ is nonnegative, $(\mathbf{c}|\succ)$ is term order. By replacing the objective function by $(\mathbf{c}|\succ)$, the optimal value of $\mathrm{IP}_{\mathbf{A},(\mathbf{c}|\succ)}(\mathbf{b})$ does not change, but $\mathrm{IP}_{\mathbf{A},(\mathbf{c}|\succ)}(\mathbf{b})$ has the unique optimal solution.

The *Conti-Traverso algorithm* that is listed in Algorithm2.4 solves $\mathrm{IP}_{\mathbf{A},(\mathbf{c}|\succ)}(\mathbf{b})$.

Algorithm 2.4. *(Conti-Traverso Algorithm for IPP).*

STEP 1: *Compute a Groebner basis* $\mathrm{G}_{(\mathbf{c}|\succ)}(I_{\mathbf{A}})$.
STEP 2: *Compute a feasible solution* $\mathbf{u}$ *of* $\mathrm{IP}_{\mathbf{A},\mathbf{c}}(\mathbf{b})$.
STEP 3: *Compute the normal form of* $\mathbf{y}^{\mathbf{u}}$ *and store result as* $\mathbf{y}^{\mathbf{v}}$.
$\mathbf{v}$ *is the unique optimal of* $\mathrm{IP}_{\mathbf{A},(\mathbf{c}|\succ)}(\mathbf{b})$.

2.5 Toric Ideals of Incidence Matrices of Complete Graphs

Next we introduce the result of [10], which plays an important role in TSP. Let K_d be the complete graph with d nodes. In this section, denote the node-edge incidence matrix of K_d by $\mathbf{A}$. Furthermore, we use the term edge for the closed line segment joining their end nodes. And we identify the nodes of K_d with the vertices of a regular d-gon in the plane labeled clockwise from 1 to d.

One result of this paper is the fact that the reduced Groebner basis of $I_{\mathbf{A}}$ has a good property, it can be obtained without applying the Buchberger's algorithm [10].

Consider following polynomial ring:

$$\mathrm{K}[\mathbf{y}] \stackrel{\text{def}}{=} \mathrm{K}[y_{ij} \ : \ 1 \leq i < j \leq d].$$

The variables y_{ij} correspond to the edge $\{i,j\}$in K_d. Define the variable ordering as:

$$y_{ij} < y_{kl} \Longleftrightarrow \begin{cases} i < k, \\ \text{or} \\ i = k \text{ and } j > l. \end{cases}$$

Then the toric ideal $I_{\mathbf{A}}$ is the kernel of the homomorphism $\Phi \ : \ \mathrm{K}[\mathbf{y}] \rightarrow \mathrm{K}[\mathbf{t}],\ y_{ij} \mapsto t_i t_j$.

Let $\succ$ be the lexicographic term order on $\mathrm{K}[\mathbf{y}]$. Given any pair of non-intersecting edges $\{i,j\}$, $\{k,l\}$ of K_d, one of the pairs $\{i,k\}$, $\{j,l\}$ or $\{i,l\}$, $\{j,k\}$ meets in a point. With the disjoint edges $\{i,j\}$, $\{k,l\}$, we associate the binomial $y_{ij}y_{kl} - y_{il}y_{jk}$ where $\{i,l\}$, $\{j,k\}$ is the intersecting pair. We denote by C the set of all binomials obtained in this fashion.

Theorem 2.5. *([10]) The set C is the reduced Groebner basis of $I_{\mathbf{A}}$ w.r.t. $\succ$.*

We define the weight w_{ij} of the variable y_{ij} as the number of edges of K_d which do not meet the edge $\{i,j\}$. We denote $\mathbf{w} \stackrel{\text{def}}{=} (w_{ij} \ : \ 1 \leq i < j \leq d)$. Then, it is known that the set C is the reduced Groebner basis of $I_{\mathbf{A}}$ w.r.t. the order induced by $\mathbf{w}$.

3 Groebner Basis Method for TSP

3.1 Our Framework for Integer Programming Problems

Our framework is one of the iterative relaxation methods. Considering $\mathrm{IP}_{\mathbf{A},\mathbf{c}}(\mathbf{b})$. Let C be the set of all constraint equations. For each $C' \subset C$, denote the corresponding coefficient matrix and right hand vector by $\mathbf{A}'$ and $\mathbf{b}'$. We call $\mathrm{IP}_{\mathbf{A}',\mathbf{c}}(\mathbf{b}')$ a relaxation problem of $\mathrm{IP}_{\mathbf{A},\mathbf{c}}(\mathbf{b})$.

The framework is based on the following strategy: choose a subset of constraints $C_0 \subset C$. Denote the corresponding coefficient matrix and right hand vector by $\mathbf{A}^0$ and $\mathbf{b}^0$. Then solve $\mathrm{IP}_{\mathbf{A}^0,\mathbf{c}}(\mathbf{b}^0)$ by the Conti-Traverso algorithm and check if the obtained optimal solution satisfies all of C. If so, stop: we have found a optimal solution. If not, chose some violated constraints and add them to C_0 and form C_1. Then solve $\mathrm{IP}_{\mathbf{A}^1,\mathbf{c}}(\mathbf{b}^1)$. Repeat this process until obtain a optimal. This framework is very natural, but good points of ours are:

(i) the solution obtained in each iteration is integral,

(ii) the Groebner basis computation of each iteration can be done by applying the Groebner Walk algorithm.

The property (i) can be guaranteed by the Conti-Traverso algorithm.

3.2 Applications for TSP

In this subsection, we introduce two algorithm for TSP as an application of our method for IPP. One is based on the iterative relaxation method, and the other is based on the branch-and-bound method.

(Algorithm 1, based on the iterative relaxation method):

Recall that there are three kind of constraints: (i) "degree constraints", (ii) "subtour elimination", and (iii) "0-1 constraints". TSP is represented as following 0-1 integer programming.

$$
\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{\{i,j\}\in E} c_{ij}x_{ij}, & \\
\text{subject to} & \displaystyle\sum_{j\,:\,\{i,j\}\in E} x_{ij} = 2, & \forall i \in V, \\
& \displaystyle\sum_{\{i,j\}\in C} x_{ij} \leq |C| - 1, & \forall C \in \mathcal{C}, \\
& x_{ij} \in \{0,1\}, & \forall \{i,j\} \in E.
\end{array}
$$

$\mathcal{C}$ is the set of all subtours in the graph. The size of the coefficient matrices for TSP are so large that it is impossible to solve with the Conti-Traverso algorithm. It is because of the amount of "subtour elimination constraints". According to our experiments, TSP on the graph with 5 nodes and more is not solvable by the Conti-Traverso algorithm. However, the coefficient matrices of TSP have several good features which are suitable for our method for IPP, based on the iterative relaxation method. The algorithm consists of 3 steps, and it is shown in Figure 3.2.

In **STEP 1** in the kth iteration, solve the integer programming $\mathrm{IP}_{\mathbf{A}_k,\mathbf{c}^k}(\mathbf{b}^k)$ with the Conti-Traverso algorithm. Remark that because we consider on the complete graph, a feasible solution for Conti-Traverso algorithm is trivial (circular tour, for example). In addition, in the 1st iteration, we can make good use of results of [?] and the Groebner Walk to compute the reduced Groebner basis.

In **STEP 2**, check the feasibility for TSP. From the integrality of the solution, it can be done on the realm of graph theory, by checking whether $\mathbf{x}^k$ forms a tour in the graph or not. If yes, it is an optimal solution for the TSP, then output it. Otherwise, goto STEP 3.

In **STEP 3**, choose a violated constraint arbitrarily, then add the constraint to $\mathrm{IP}_{\mathbf{A}_k,\mathbf{c}^k}(\mathbf{b}^k)$. It can be done by expanding $\mathbf{A}_k$, $\mathbf{b}$ and $\mathbf{c}^k$. Note that, the new constraint is represented as a inequality, then the slack

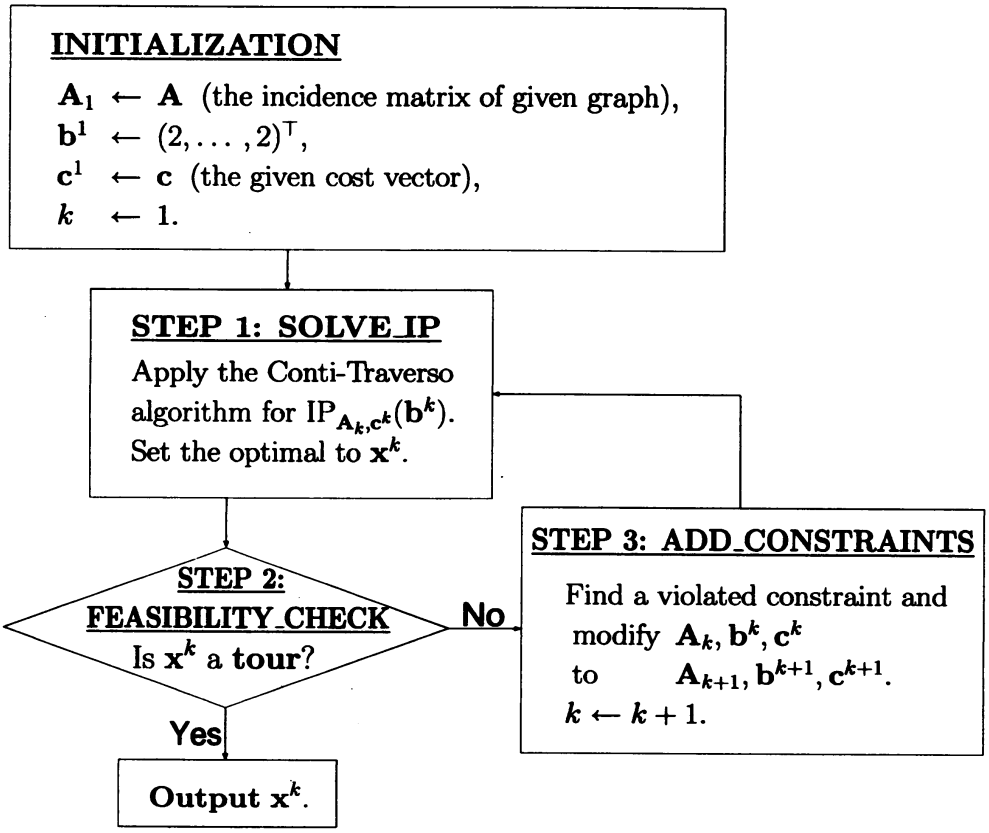

図 1: The framework of the algorithm for TSP

variable must be added. The costs corresponding to slack variables are set to zero (in the fact, any value is available).

(Algorithm 2, based on the Branch-and-Bound method):

In TSP, branch-and-bound method can be applied as follow: let $\bar{C}$ be the initial trivial feasible solution and $\bar{q}$ be its cost(value of the objective function). First, solve the problem in which only (i)degree constraints and (iii)$0-1$ constraints are considered, denote by $\mathcal{P}(\emptyset,\emptyset)$. If there is no subtours in the solution, the we obtain the optimal solution. If not, choose a subtour arbitrarily and denote the edges on it by $C = \{l_1, l_2, \ldots, l_r\}$ along the tour (i.e., l_i and l_{i+1} $(i = 1,\ldots,r-1)$ are adjacent). The problem $\mathcal{P}(\emptyset,\emptyset)$ can be divided into r subproblems with additional constraints:

$\mathcal{P}(\{l_1,\ldots,l_{r-1}\},\{l_r\})$: fix $l_1,\ldots,l_{r-1}$ to use, l_r not to use,
$\mathcal{P}(\{l_1,\ldots,l_{r-2}\},\{l_{r-1}\})$: fix $l_1,\ldots,l_{r-2}$ to use, l_{r-1} not to use,
...
$\mathcal{P}(\{l_1\},\{l_2\})$: fix l_1 to use, l_2 not to use,
$\mathcal{P}(\emptyset,\{l_1\})$: fix l_1 not to use.

It is easy to see that the optimal solution of TSP is a solution of a subproblem. Solve a subproblem $\mathcal{P}(\{l_1,\ldots,l_{j-1}\},\{l_j\}), 1 \le j \le r$, then three cases can be occurred:

(case 1) obtained cost is greater than $\bar{q}$,

(case 2) obtained cost is not greater than $\bar{q}$ and the solution forms a tour,

(case 3) obtained cost is not greater than $\bar{q}$ and the solution does not form a tour.

case 1. The feasible region of this subproblem never contains the optimal solution. Exit the subproblem.
case 2. We obtain a better solution than $\bar{C}$. Replace $\bar{C}$ by this solution and $\bar{q}$ by its cost, and exit the subproblem.
case 3. We obtain the subtour with form $\{l_1, \ldots, l_{j-1}, \tilde{l}_1, \ldots, \tilde{l}_s\}$. Then we divide into the subproblems again and solve them:

$$\mathcal{P}(\{l_1, \ldots, l_j, \tilde{l}_1, \ldots, \tilde{l}_{s-1}\}, \{\tilde{l}_s\}),$$
$$\mathcal{P}(\{l_1, \ldots, l_j, \tilde{l}_1, \ldots, \tilde{l}_{s-2}\}, \{\tilde{l}_{s-1}\}),$$
$$\ldots$$
$$\mathcal{P}(\{l_1, \ldots, l_j, \tilde{l}_1\}, \{\tilde{l}_2\}),$$
$$\mathcal{P}(\{l_1, \ldots, l_j\}, \{\tilde{l}_1\}).$$

It is easy to see that this algorithm must terminate after solving finite subproblems, and $\bar{C}$ is the optimal solution. We talk about the advantage of applying the Conti-Traverso algorithm to this branch-and-bound framework of TSP. Let A be the incidence matrix of the complete graph with d nodes. Moreover, let $\mathbf{x}$ and $\mathbf{c}$ be the variables and cost vector, and n be the number of edges.

The problem $\mathcal{P}(\emptyset, \emptyset)$ can be represented by following integer programming problem:

$$\left|\begin{array}{ll} \min & \mathbf{c} \cdot \mathbf{x} \\ \text{s.t.} & \begin{pmatrix} A & \mathbf{0} \\ \mathbf{I} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{1} \\ \mathbf{2} \end{pmatrix} \\ & \mathbf{x} \in \{0,1\}^n \\ & \mathbf{y} \in \{0,1\}^n \end{array}\right. \tag{3.1}$$

where $\mathbf{y}$ is the slack variables, $\mathbf{I}$ is n-dimensional identity matrix, and $\mathbf{0}$ is $d \times n$-zero matrix. An important fact is that the coefficient matrix is Lawrence type. Hence the reduced Groebner basis of its toric ideal is the minimal universal Groebner basis simultaneously (i.e., a Groebner basis for all term order). This is very good feature when consider subproblems. In subproblems, there are additional constraints on edge, "must be used" and "must not be used". The property "must be used" can be expressed by setting its cost to $-M$, where M is sufficient big number (like "big-M method" to linear programming problems). The property "must not be used" can be expressed by setting its cost to M. Assume that the reduced Groebner basis $\mathcal{G}$ of the coefficient matrix of 3.1 had been obtained. Hence $\mathcal{G}$ is a universal Groebner basis, we can solve subproblems with the Conti-Traverso algorithm by modifying the cost vector $\mathbf{c}$ adequately: find normal form with respect to the modified cost vector.

One problem is that the order induced by the modified cost vector may not be a term order. But following two lemmas guarantee its correctness.

Lemma 3.1 ([19], Lemma 4.14): *Let $\mathcal{A}$ be a $d \times n$-integral matrix. The toric ideal I_A is homogeneous if and only if there exists a vector $\omega \in \mathbb{Q}^d$ such that $\mathbf{a}_i \cdot \omega = 1$ for all $i = 1, \ldots, n$.*

Clearly, the toric ideal of the coefficient matrix of 3.1 is homogeneous($\omega = (0, \ldots, 0, 1, \ldots, 1)$).

Lemma 3.2 ([19], Theorem 2.5): *The Groebner fan* $\mathrm{GF}(I)$ *is complete if I is homogeneous with respect to some positive grading.*

Thanks to above, using the order induced by the modified cost vector, division algorithm works well while it is not a term order.

4 Computational Experience and Conclusions

We implemented our algorithms on RISA/ASIR with some OpenXM libraries. Computational experiences ware done on Linux machine with AMD-K6(400MHz) and 384MB RAM.

In this paper, two approaches for TSP based on the Conti-Traverso algorithm was proposed. We combine the iterative methods (Algorithm 1) and the Branch-and-Bound method (Algorithm 2) with the Conti-Traverso algorithm. The computing result is listed in Table 1. They are very slow because of the Groebner basis computation.

表 1: Use our framework.

nodes	5	6	7	8	9	10	11	12
Algorithm 1(sec.)	1.4	7.7	150	2600	7500	93000	-	-
Algorithm 2(sec.)	9.0	240	7100	-	-	-	-	-

From this result, we cannot insist that the Groebner basis is a tool to solve the IPP. But it is a tool to analyse combinatorial optimization problems from the view of the polynomial ring and ideals.

参考文献

[1] W. W. Adams and P. Loustaunau. *An Introduction to Gröbner Bases*, volume 3 of *Graduate Studies in Mathematics*. American Mathematical Society, 2nd edition, 1996.

[2] T. Becker and V. Weispfenning. *Gröbner Bases*, volume 141 of *Graduate Texts in Mathematics*. Springer, 2nd edition, 1998.

[3] C. Berge. *Graphs*. North-Holland, 2nd edition, 1985.

[4] A. M. Bigatti, R. Scala, and L. Robbiano. Computing toric ideals. *Journal of Symbolic Computation*, 27:351–365, 1999.

[5] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. The Macmillan Press LTD, 1976.

[6] B. Buchberger. *Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory*, chapter 6, pages 184–232. D. Riedel Publishing Company, Dordrecht, 1985. edited by N. K. Bose.

[7] S. Collart, M. Kalkbrener, and D. Mall. Converting bases with the Gröbner walk. *Journal of Symbolic Computation*, 24:465–469, 1997.

[8] P. Conti and C. Traverso. Gröbner bases and integer programming. In *Proceedings AAECC-9(New Orleans)*, volume 539 of *LNCS*, pages 130–139. Springer Verlag, 1991.

[9] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms*. Undergraduate Texts in Mathematics. Springer, 2nd edition, 1998.

[10] J. A. de Loera, B. Sturmfels, and R. R. Thomas. Gröbner bases and triangulations of the second hypersimplex. *Combinatorica*, 15(3):409–424, 1995.

[11] D. Eisenbud. *Commutative Algebra with a View Toward Algebraic Geometry*, volume 150 of *Graduate Texts in Mathematics*. Springer, New York, 1994.

[12] J. C. Faugère. A new efficient algorithm for computing gröbner bases (f_4). *Journal of Pure and Applied Algebra*, 139:61–88, 1999.

[13] J. C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16:329–344, 1993.

[14] D. Kapur and T. Saxena. An algorithm for converting a degree Gröbner basis to a lexicographic Gröbner basis. Rough Draft.

[15] E. Lawler, J. Lenstra A. Rinnooy Kan, and D. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization.* John Wiley & Sons, 1985.

[16] L. Pottier. Gröbner bases of toric ideals. Technical Report 2224, National De Recherche en Informatique et Automatique, 3 1994.

[17] A. Schrijver. *Theory of Linear and Integer Programming.* John Willey & Sons, 1986.

[18] B. Sturmfels. Gröbner bases of toric varieties. *Tôhoku Mathematical Journal*, 43:249–261, 1991.

[19] B. Sturmfels. *Gröbner Bases and Convex Polytopes*, volume 8 of *University Lecture Series.* American Mathematical Society, 1995.

[20] B. Sturmfels and R. R. Thomas. Variation of cost functions in integer programming. *Mathematical Programming*, 77:357–387, 1997.

[21] R. R. Thomas. A geometric Buchberger algorithm for integer programming. *Mathematics of Operations Research*, 20:864–884, 1995.

[22] R. R. Thomas. Applications to integer programming. In *Proceedings of Symposia in Applied Mathematics*, volume 53, 1998.

[23] R. R. Thomas. *Gröbner Bases in Integer Programming.* Kluwer Academic Publishing, 1998. Handbook of Combinatorial Optimization.

[24] C. Traverso. Hilbert functions and the Buchberger algorithm. *Journal of Symbolic Computation*, 22:355–376, 1996.